ITW

2131

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/841,503	04/24/2001	Richard Alan Dayan	RPS9 2001 0011	5669

53493 7590 10/17/2005

LENOVO (SINGAPORE) PTE. LTD.
BUILDING 675, MAIL C-137
4401 SILICON DRIVE
DURHAM, NC 27709

RECEIVED
OIPE/IAP
OCT 31 2005

EXAMINER	
HENNING, MATTHEW T	
ART UNIT	PAPER NUMBER
2131	

DATE MAILED: 10/17/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

Office Action Summary	**Application No.** 09/841,503	**Applicant(s)** DAYAN ET AL.
	Examiner Matthew T. Henning	**Art Unit** 2131

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *01 July 2005*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *24 April 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

This action is in response to the communication filed on 7/1/2005.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 7/1/2005 has been entered.

Response to Arguments

Applicants' arguments filed 7/1/2005 have been fully considered but they are not persuasive. Applicants argues primarily that:

a. Gafken and Hasbun do not disclose comparing the similarity of the update portion with the protected partition.

b. Gafken and Hasbun did not disclose locking a protected partition in a hard drive.

c. Schneier did not teach storing the random password in a database at the server.

Regarding applicants' argument a. that Gafken and Hasbun do not disclose comparing the similarity of the update portion with the protected partition, the examiner has considered the argument and does not find the argument persuasive. Hasbun clearly teaches that the update file contains objects and the BIOS is searched for previous versions of the object (See Hasbun Col. 12 Line 59 – Col. 13 Line 17. Also see Col. 13 Line 18 – Col. 16 Line 27). This required a comparison of similarity in order to determine whether the object already existed. As such, the examiner does not find the argument persuasive.

Regarding applicants' argument b. that Gafken and Hasbun do not disclose locking a protected partition in a hard drive, the examiner has considered the argument and does not find the argument persuasive. Gafken disclosed locking the blocks of data after modification was performed (See Gafken Col. 13 Paragraph 9 – Col. 14 Paragraph 1). Furthermore, as discussed below in the rejection of claim 1 under 35 USC 103(a), it was well known in the art that a hard drive could be used in place of flash memory to store information including a BIOS. As such, it would have been obvious to replace the flash memory of Gafken and Hasbun with a partition on a hard drive. As such, the examiner does not find the argument persuasive.

Applicants' argument c. has been considered but is moot in view of the new ground(s) of rejection. See below.

All rejections and objections not specifically set forth below have been withdrawn.

Claims 1-30 have been examined, and claims 31-36 have been cancelled.

Claim Rejections - 35 USC § 112

The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

Claims 1-30 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

The term "similar" in claims 1, 11, 13, and 26 is a relative term which renders the claim indefinite. The term "similar" is not defined by the claim, the specification does not provide a standard for ascertaining the requisite degree, and one of ordinary skill in the art would not be reasonably apprised of the scope of the invention. One of ordinary skill in the art would be unable to determine how alike the information in the protected partition and the portion of information stored in the update file would need to be in order to be considered "similar" to each other. As such, the ordinary person skilled in the art would be unable to determine the scope of the claim. Therefore, claims 1-30 are rejected for failing to particularly point out and distinctly claim the subject matter which the applicants regard as the invention.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1- 4,13-19, and 26-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Gafken (US Patent Number 6,026,016), and further in view of Hasbun et al. (U.S. Patent Number 6,088,759) hereinafter referred to as Hasbun.

Regarding claim 1, Gafken disclosed a method for updating a protected partition within a hard drive of a computing system, wherein said method comprises (See Gafken Fig. 5): starting execution of an initialization program in a processor within said computing system in response to turning on electrical power within said computing system (See Gafken Col. 3 Paragraph 2 Lines 1-4); determining whether an update partition file is stored in non-volatile storage (See Gafken Col. 5 Paragraph 5) within said computing system for subsequently updating said protected partition (See Gafken Col. 13 Paragraphs 4 and 7); after determining that said update partition is stored within said computing system for updating said protected partition, writing a portion of said update partition file to said protected partition (See Gafken Col. 13 Paragraph 8); and locking said protected partition to prevent further modification of information stored within said protected partition (See Gafken Col. 13 Paragraph 9 – Col. 14 Paragraph 1), but failed to disclose overwriting similar parts and appending new parts.

Hasbun teaches that a bios update can be allocated into virtual blocks so that the blocks can be updated individually without having to erase the entire memory first (See Hasbun Col. 5 Paragraph 6 – Col. 6 Paragraph 2 and Col. 12 Line 59 – Col. 16 Line 27). Hasbun also teaches that new blocks should be allocated from existing free memory (See Hasbun Col. 7 Paragraph 2).

It would have been obvious to the ordinary person skilled in the art at the time of invention to employ the teachings of Hasbun to the bios updating system of Gafken by updating each update part one at a time. This would have been obvious because the ordinary person skilled in the art would have been motivated to provide a safe method for updating a bios without risking loss of the entire bios in the event of a power failure.

Furthermore, it was well know at the time of the invention that a hard drive could be used in place of flash memory, even to store a BIOS. As such, it would have been obvious to the ordinary person skilled in the art at the time of invention to employ what was well known in the art at the time of invention in the BIOS system of Gafken and Hasbun by storing the BIOS in a hard drive instead of a flash memory. This would have been obvious because the ordinary person skilled in the art would have been motivated to provide greater storage capacity for the BIOS and to make updating the BIOS fast and efficient.

Regarding claim 13, the combination of Gafken and Hasbun disclosed a method for updating a protected partition within a hard drive of a client computing system, wherein said method comprises: generating an update partition file within a server (See Gafken Col. 12 Paragraph 7 – Col. 13 paragraph 1, wherein it was inherent that the server created the image by signing it in order for the server to be verified through digital signatures); transferring said update partition file from said server to said client computing system (See Gafken Col. 12 Paragraph 5); storing said update partition file in non-volatile storage within said client computing system (See Gafken Col. 5 Paragraph 5); starting execution of an initialization program in a processor within said client computing system in response to turning on electrical power within said client computing system (See Gafken Col. 3 Paragraph 2 Lines 1-4); determining that said update partition file is stored in non-volatile storage within said client computing system (See Gafken Col. 13 Paragraphs 4 and 7); writing a portion of said update partition file to said protected partition (See Gafken Col. 13 Paragraph 8); and locking said protected partition to prevent further modification of information stored within said protected partition (See Gafken Col. 13 Paragraph 9 – Col. 14 Paragraph 1). The combination of Gafken and Hasbun further disclosed comparing information stored in said protected partition with information within said update partition file; when a matching portion of said information stored in said protected partition is found to be similar to said entry, said matching portion is overwritten with said entry if space around said matching portion is sufficient, and when a matching portion of said information stored in said protected partition is not found to be similar to said entry, said entry is appended to said information stored in said protected partition if space within said protected partition is sufficient (See the rejection of claim 1 above).

Claim 26 recites a computer system comprising: a processor executing an initialization program in response to power being turned on in said computer program (See Gafken Fig. 1

Element 110); a hard drive having a protected partition blocked during execution of an initialization program to prevent changing information stored within said protected partition (See Fig. 1 Element 130); non-volatile storage storing an update partition data structure for modifying contents of said protected partition and said initialization program, wherein said initialization program executing within said processor determines that said update partition data structure is stored in said non-volatile storage, writes a portion of said update partition data structure to said protected partition, and locks said protected partition to prevent further modification of information stored within said protected partition (See rejection of claim 1 above). The combination of Gafken and Hasbun further disclosed comparing information stored in said protected partition with information within said update partition file; when a matching portion of said information stored in said protected partition is found to be similar to said entry, said matching portion is overwritten with said entry if space around said matching portion is sufficient, and when a matching portion of said information stored in said protected partition is not found to be similar to said entry, said entry is appended to said information stored in said protected partition if space within said protected partition is sufficient (See the rejection of claim 1 above).

Regarding claims 2, 17, and 27, the combination of Gafken and Hasbun disclosed that a flag bit is set in non-volatile storage within said computing system when said update partition file is stored at a predetermined location in non-volatile storage within said computing system (See Gafken Col. 13 Paragraphs 3-4), and determining whether said update partition is stored within said computing system for updating said protected partition is performed by determining whether said flag bit is set (See Gafken Col. 13 Paragraph 7 and Fig. 5 Step 550).

Regarding claims 3, 18, and 28, the combination of Gafken and Hasbun disclosed that after determining that said update partition file is stored within said computing system for updating said protected partition, verifying whether said update partition file has been generated by a trusted server system, and said portion of said update partition is written to said protected partition only following verification that said update partition file has been generated by a trusted server system (See Gafken Col. 12 Paragraph 6 – Col. 13 Paragraph 1 and Figure 6).

Regarding claim 4, the combination of Gafken and Hasbun disclosed that verification that said update partition file has been generated by said trusted server system includes: forming a first message digest by applying a hash algorithm to a portion of said update partition file; forming a second message digest by decrypting a digital signature within said update partition file using a public key of said trusted server system; and determining that said first and second message digests are identical (See Gafken Col. 12 Paragraph 7 Line 10 – Col. 13 Line 2).

Regarding claim 14, the combination of Gafken and Hasbun disclosed that the update partition file is transferred from said server to said client computing system by means of electrical signals transmitted through a public switched telephone network (See Gafken Col. 4 Paragraph 7 wherein it was inherent that the update file was received through the wireless transmitter, and therefore through a public switched telephone network).

Regarding claim 15, the combination of Gafken and Hasbun disclosed that update partition file is transferred from said server to said client computing system by means of electrical signals transmitted over a local area network (See Gafken Col. 12 Paragraph 5).

Regarding claim 16, the combination of Gafken and Hasbun disclosed that transferring said update partition file from said server to said client computing system includes: writing said update partition file to a removable computer readable medium from said server; transporting said removable computer readable medium from said sever to said client computing system; and reading said update partition file from said removable computer readable medium into said client computing system (See Gafken Col. 12 Paragraph 5 wherein it was inherent that the image was stored to a floppy disk and retrieved from the floppy disk in order for the image to have been obtained through a floppy drive).

Regarding claim 19, the combination of Gafken and Hasbun disclosed the use of digital signatures to verify the origin of the update file (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Claims 5, 6, 20-21, and 33 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Gafken and Hasbun as applied to claims 3 and 18 above, and further in view of Menezes et al. ("Handbook of Applied Cryptography") hereinafter referred to as Menezes.

Regarding claims 5 and 20, the combination of Gafken and Hasbun disclosed the use of digital signatures, including public and private keys, in order to verify that a valid server generated the boot image (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1), but the combination of Gafken and Hasbun failed to disclose the use of a password in the signature. However, the combination of Gafken and Hasbun did disclose the use of password challenges.

Menezes teaches that providing a sequence number (password), stored and updated at both a receiver and a sender, in a digital signature of the sender, protects the signature against replay attacks (See Menezes Page 399 Section (ii).

It would have been obvious to the ordinary person skilled in the art at the time of invention to employ the teachings of Menezes to the validation signatures of the combination of Gafken and Hasbun by providing a sequence number in the signature of the update image. This would have been obvious because the ordinary person skilled in the art would have been motivated to provide protection against illicitly signed updates.

Regarding claims 6 and 21, the combination of Gafken, Hasbun, and Menezes disclosed that the data includes said version of said setup password appended to a portion of said update partition file (See rejection of claim 5 above), said algorithm is a hash algorithm generating a message digest (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1), and verifying that said update partition file has been generated by said trusted server system includes applying said hash algorithm to said setup password stored within said computing system appended to a portion of said update partition file to generate a first version of a message digest and comparing said first version of said message digest with a second version of said message digest obtained by signing said encrypted portion of said update partition file (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Claims 7, 8, 11, 22, 23, and 29 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Gafken and Hasbun as applied to claims 1, 13, and 28 above, and further in view of Hayashi et al. (US 2001/0039651 A1) hereinafter referred to as Hayashi.

Regarding claims 7, 22, and 29, the combination of Gafken and Hasbun disclosed digitally signing the update file and verifying the signature prior to updating the partition (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1), but the combination of Gafken and Hasbun failed to disclose encrypting portions of the file separately and verifying each portion individually.

Hayashi teaches a method for providing a variety of software safely by breaking the file into pieces and decrypting each piece separately (See Hayashi Page 1 Col. 2 Paragraphs 3-10).

It would have been obvious to the ordinary person skilled in the art at the time of invention to employ the teachings of Hayashi to the updating system of the combination of Gafken and Hasbun by encrypting parts of the file separately from the other parts. This would have been obvious because the ordinary person skilled in the art would have been motivated to provide users with customized software without imposing too much of a load on the provider. In this combination, it would also be obvious that each block contained information to be stored in a different location from the other blocks. This would have been obvious because the ordinary person skilled in the art would have been motivated not perform unnecessary computation during the update.

Regarding claim 8, the combination of Gafken, Hasbun, and Hayashi disclosed forming a first message digest by applying a hash algorithm to said entry, and forming a second message digest by signing said encrypted element associated with said entry using a public key of said trusted server system, and determining that said first and second message digests are identical (See Gafken Col. 12 Paragraph 7 Line 10 – Col. 13 Line 2).

Regarding claim 11, the combination of Gafken, Hasbun, and Hayashi disclosed that information stored in said protected partition is compared to each entry in said plurality of entries within said update partition, when a matching portion of said information stored in said protected partition is found to be similar to said entry, said matching portion is overwritten with said entry if space around said matching portion is sufficient, and when a matching portion of said information stored in said protected partition is not found to be similar to said entry, said entry is appended to said information stored in said protected partition if space within said protected partition is sufficient (See the rejection of claim 1 above).

Regarding claim 23, the combination of Gafken, Hasbun, and Hayashi disclosed that each encrypted element is formed in said server by applying a hash algorithm to said entry, forming a first message digest, and by signing said first message digest with a private key of said server; and verification that said entry has been generated by said server includes forming a second message digest by applying a hash algorithm to said entry, forming a third message digest by signing said encrypted element associated with said entry using a public key of said server, and determining that said second and third message digests are identical (See Gafken Col. 12 Paragraph 7 Line 10 – Col. 13 Line 2).

Claims 9, 10, 24-25, 30-32, and 34-35 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Gafken, Hasbun, and Hayashi as applied to claims 7, 22 and 29 above, and further in view of Menezes.

Regarding claim 9, Gafken, Hasbun and Hayashi disclosed the use of digital signatures, including public and private keys, in order to verify that a valid server generated the boot image parts (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1), but Gafken, Hasbun, and Hayashi did not disclose the use of a password in the signature. However, Gafken, Hasbun and Hayashi did disclose the use of password challenges (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Menezes teaches that providing a sequence number (password), stored and updated at both a receiver and a sender, in a digital signature of the sender, protects the signature against replay attacks (See Menezes Page 399 Section (ii).

It would have been obvious to the ordinary person skilled in the art at the time of invention to employ the teachings of Menezes to the validation signatures of the combination of Gafken and Hasbun by providing a sequence number in the signature of the update image. This would have been obvious because the ordinary person skilled in the art would have been motivated to provide protection against illicitly signed updates.

Regarding claim 10, the combination of Gafken, Hasbun, Hayashi, and Menezes disclosed that the data includes said version of said setup password appended to a said entry (See rejection of claim 5 above), said algorithm is a hash algorithm generating a message digest, and verifying that said entry has been generated by said trusted server system includes applying said hash algorithm to said setup password stored within said computing system appended said entry to generate a first version of a message digest and comparing said first version of said message digest with a second version of said message digest obtained by signing said encrypted element (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Regarding claim 24, the combination of Gafken, Hasbun, Hayashi, and Menezes disclosed that a setup password is stored in non-volatile storage within said client computing system; a copy of said setup password is stored in a database accessed by said Server (See rejection of claim 5 above); said encrypted element of said update partition file is prepared in said server by signing, with a private key of said server, a result of the application of an algorithm to data including said copy of said setup password', and verification within said client computing system that said entry has been generated by said server includes signing said encrypted element associated with said entry with said public key of said server (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Claim 25 is rejected for the same reasons as claim 10 above as applied to claim 24 above.

Regarding claim 30, the combination of Gafken, Hasbun, Hayashi, and Menezes disclosed that the non-volatile storage additionally stores a setup password, and each said encrypted element includes a digital signature signed by said trusted server system, wherein said digital signature is formed by applying a hash algorithm to an entry associated with said encrypted element to form a message digest and by signing said message digest with a private key of said trusted server system (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Claim 12 is rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Gafken and Hasbun as applied to claim 1 above, and further in view of Schmidt (U.S. Patent Number 5,826,015).

The combination of Gafken and Hasbun disclosed a secure bios updating system (See rejection of claim 1 above) but failed to disclose requiring a user to input a password to unlock the bios write capabilities. However, Gafken and Hasbun did disclose the use of password challenges (See Gafken Col. 12 Paragraph 7 – Col. 13 Paragraph 1).

Schmidt teaches that in order to remotely upgrade a bios, an administrator password should be provided in order to unlock the partition (See Schmidt Fig. 9 and abstract).

It would have been obvious to the ordinary person skilled in the art at the time of invention to employ the teachings of Schmidt to the bios updating system of Gafken by requiring a correct password to be entered in order to unlock the bios altering capabilities. This would have been obvious because the ordinary person skilled in the art would have been motivated to protect the current bios from accidental or illicit alterations.

Conclusion

Claims 1-30 have been rejected, and claims 31-36 have been cancelled.

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

i. Arnold et al. (US Patent Number 5.128.995) disclosed a system in which a BIOS was stored in a locked portion of a Hard Drive in order to update the BIOS more easily.

ii. Harmer (US Patent Number 5,835,760) disclosed a system which stores a BIOS in a Hard Drive and searches for portions in the BIOS to update.

iii. Zinger et al. (US Patent Number 6,836,847) disclosed that a Hard Drive could be used in place of Flash Memory.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Matthew T. Henning whose telephone number is (571) 272-3790. The examiner can normally be reached on M-F 8-4.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Ayaz Sheikh can be reached on (571) 272-3795. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

[illegible]
Matthew Henning
Assistant Examiner
Art Unit 2131
9/29/2005

[illegible]
AYAZ SHEIKH
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100

Notice of References Cited	Application/Control No. 09/841,503	Applicant(s)/Patent Under Reexamination DAYAN ET AL.
	Examiner Matthew T. Henning	Art Unit 2131 / Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	Classification
	A	US-5,128,995	07-1992	Arnold et al.	713/1
	B	US-5,835,760	11-1998	Harmer, Tracy D.	713/2
	C	US-6,836,847	12-2004	Zinger et al.	726/19
	D	US-5,966,541	10-1999	Agarwal, Anant	717/132
	E	US-2001/0044782	11-2001	Hughes et al.	705/59
	F	US-			
	G	US-			
	H	US-			
	I	US-			
	J	US-			
	K	US-			
	L	US-			
	M	US-			

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	Classification
	N					
	O					
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	Menezes et al. "Handbook of Applied Cryptography", 1997, CRC Press, pp 397-405
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

HANDBOOK of APPLIED CRYPTOGRAPHY

Alfred J. Menezes
Paul C. van Oorschot
Scott A. Vanstone

CRC

CRC Press
Boca Raton London New York Washington, D.C.

Library of Congress Cataloging-in-Publication Data

Menezes, A. J. (Alfred J.), 1965–
Handbook of applied cryptography / Alfred Menezes, Paul van Oorschot, Scott Vanstone.
p. cm. -- (CRC Press series on discrete mathematics and its applications)
Includes bibliographical references and index.
ISBN 0-8493-8523-7 (alk. paper)
1. Computers--Access control--Handbooks, manuals, etc.
2. Cryptography--Handbooks, manuals, etc. I. Van Oorschot, Paul C.
II. Vanstone, Scott A. III. Title. IV. Series: Discrete mathematics and its applications.
QA76.9.A25M463 1996
0005.8'2--dc21 96-27609
CIP

International Standard Book Number 0-8493-8523-7
Library of Congress Card Number 96-27609
Printed in the United States of America 5 6 7 8 9 0
Printed on acid-free paper

10.7 Note (*pre-play attack*) Protocol 10.6 and similar one-time password schemes including that of Note 10.8 remain vulnerable to an active adversary who intercepts and traps (or impersonates the system in order to extract) an as-yet unused one-time password, for the purpose of subsequent impersonation. To prevent this, a password should be revealed only to a party which itself is known to be authentic. Challenge-response techniques (see §10.3) address this threat.

10.8 Note (*alternative one-time password scheme*) The following one-time-password alternative to Protocol 10.6 is suitable if storing actual passwords on the system side is acceptable (cf. Figure 10.1; compare also to §10.3.2(iii)). The claimant A has a shared password P with the system verifier B, to which it sends the data pair: $(r, H(r, P))$. The verifier computes the hash of the received value r and its local copy of P, and declares acceptance if this matches the received hash value. To avoid replay, r should be a sequence number, timestamp, or other parameter which can be easily guaranteed to be accepted only once.

10.3 Challenge-response identification (strong authentication)

The idea of cryptographic challenge-response protocols is that one entity (the claimant) "proves" its identity to another entity (the verifier) by demonstrating knowledge of a secret known to be associated with that entity, without revealing the secret itself to the verifier during the protocol.[3] This is done by providing a response to a time-variant challenge, where the response depends on both the entity's secret and the challenge. The *challenge* is typically a number chosen by one entity (randomly and secretly) at the outset of the protocol. If the communications line is monitored, the response from one execution of the identification protocol should not provide an adversary with useful information for a subsequent identification, as subsequent challenges will differ.

Before considering challenge-response identification protocols based on symmetric-key techniques (§10.3.2), public-key techniques (§10.3.3), and zero-knowledge concepts (§10.4), background on time-variant parameters is first provided.

10.3.1 Background on time-variant parameters

Time-variant parameters may be used in identification protocols to counteract replay and interleaving attacks (see §10.5), to provide uniqueness or timeliness guarantees, and to prevent certain chosen-text attacks. They may similarly be used in authenticated key establishment protocols (Chapter 12), and to provide uniqueness guarantees in conjunction with message authentication (Chapter 9).

Time-variant parameters which serve to distinguish one protocol instance from another are sometimes called *nonces*, *unique numbers*, or *non-repeating values*; definitions of these terms have traditionally been loose, as the specific properties required depend on the actual usage and protocol.

10.9 Definition A *nonce* is a value used no more than once for the same purpose. It typically serves to prevent (undetectable) replay.

[3] In some mechanisms, the secret is known to the verifier, and is used to verify the response; in others, the secret need not actually be known by the verifier.

The term *nonce* is most often used to refer to a "random" number in a challenge-response protocol, but the required randomness properties vary. Three main classes of time-variant parameters are discussed in turn below: random numbers, sequence numbers, and timestamps. Often, to ensure protocol security, the integrity of such parameters must be guaranteed (e.g., by cryptographically binding them with other data in a challenge-response sequence). This is particularly true of protocols in which the only requirement of a time-variant parameter is uniqueness, e.g., as provided by a never-repeated sequential counter.[4]

Following are some miscellaneous points about time-variant parameters.

1. Verifiable timeliness may be provided through use of random numbers in challenge-response mechanisms, timestamps in conjunction with distributed timeclocks, or sequence numbers in conjunction with the maintenance of pairwise (claimant, verifier) state information.
2. To provide timeliness or uniqueness guarantees, the verifier in the protocol controls the time-variant parameter, either directly (through choice of a random number) or indirectly (through information maintained regarding a shared sequence, or logically through a common time clock).
3. To uniquely identify a message or sequence of messages (protocol instance), nonces drawn from a monotonically increasing sequence may be used (e.g., sequence or serial numbers, and timestamps, if guaranteed to be increasing and unique), or random numbers of sufficient size. Uniqueness is often required only within a given key lifetime or time window.
4. Combinations of time-variant parameters may be used, e.g., random numbers concatenated to timestamps or sequence numbers. This may guarantee that a pseudorandom number is not duplicated.

(i) Random numbers

Random numbers may be used in challenge-response mechanisms, to provide uniqueness and timeliness assurances, and to preclude certain replay and interleaving attacks (see §10.5, including Remark 10.42). Random numbers may also serve to provide unpredictability, for example, to preclude chosen-text attacks.

The term *random numbers*, when used in the context of identification and authentication protocols, includes pseudorandom numbers which are unpredictable to an adversary (see Remark 10.11); this differs from randomness in the traditional statistical sense. In protocol descriptions, "choose a random number" is usually intended to mean "pick a number with uniform distribution from a specified sample space" or "select from a uniform distribution".

Random numbers are used in challenge-response protocols as follows. One entity includes a (new) random number in an outgoing message. An incoming message subsequently received (e.g., the next protocol message of the same protocol instance), whose construction required knowledge of this nonce and to which this nonce is inseparably bound, is then deemed to be *fresh* (Remark 10.10) based on the reasoning that the random number links the two messages. The non-tamperable binding is required to prevent appending a nonce to an old message.

Random numbers used in this manner serve to fix a relative point in time for the parties involved, analogous to a shared timeclock. The maximum allowable time between protocol messages is typically constrained by a *timeout period*, enforced using local, independent countdown timers.

[4]Such predictable parameters differ from sequence numbers in that they might not be bound to any stored state. Without appropriate cryptographic binding, a potential concern then is a pre-play attack wherein an adversary obtains the response before the time-variant parameter is legitimately sent (see Note 10.7).

e-response
me-variant
. and time-
st be guar-
e-response
: of a time-
il counter.[4]

challenge-
ocks, or se-
nt, verifier)

:ol controls
number) or
or logically

ce), nonces
ience or se-
. or random
en key life-

mbers con-
pseudoran-

uniqueness
(see §10.5,
tability, for

authentica-
n adversary
nse. In pro-
k a number
form distri-

ie entity in-
subsequen-
se construc-
und, is then
imber links
ing a nonce

r the parties
en protocol
ndependent

iy stored state.
i an adversary

10.10 Remark (*freshness*) In the context of challenge-response protocols, *fresh* typically means recent, in the sense of having originated subsequent to the beginning of the current protocol instance. Note that such freshness alone does not rule out interleaving attacks using parallel sessions (see §10.5).

10.11 Remark (*birthday repetitions in random numbers*) In generating pseudorandom numbers for use as time-variant parameters, it suffices if the probability of a repeated number is acceptably low and if numbers are not intentionally reused. This may be achieved by selecting the random value from a sufficiently large sample space, taking into account coincidences arising from the birthday paradox. The latter may be addressed by either using a larger sample space, or by using a generation process guaranteed to avoid repetition (e.g., a bijection), such as using the counter or OFB mode of a block cipher (§7.2.2).

10.12 Remark (*disadvantages of random numbers*) Many protocols involving random numbers require the generation of cryptographically secure (i.e., unpredictable) random numbers. If pseudorandom number generators are used, an initial seed with sufficient entropy is required. When random numbers are used in challenge-response mechanisms in place of timestamps, typically the protocol involves one additional message, and the challenger must temporarily maintain state information, but only until the response is verified.

(ii) Sequence numbers

A sequence number (serial number, or counter value) serves as a unique number identifying a message, and is typically used to detect message replay. For stored files, sequence numbers may serve as *version numbers* for the file in question. Sequence numbers are specific to a particular pair of entities, and must explicitly or implicitly be associated with both the originator and recipient of a message; distinct sequences are customarily necessary for messages from A to B and from B to A.

Parties follow a pre-defined policy for message numbering. A message is accepted only if the sequence number therein has not been used previously (or not used previously within a specified time period), and satisfies the agreed policy. The simplest policy is that a sequence number starts at zero, is incremented sequentially, and each successive message has a number one greater than the previous one received. A less restrictive policy is that sequence numbers need (only) be monotonically increasing; this allows for lost messages due to non-malicious communications errors, but precludes detection of messages lost due to adversarial intervention.

10.13 Remark (*disadvantages of sequence numbers*) Use of sequence numbers requires an overhead as follows: each claimant must record and maintain long-term pairwise state information for each possible verifier, sufficient to determine previously used and/or still valid sequence numbers. Special procedures (e.g., for resetting sequence numbers) may be necessary following circumstances disrupting normal sequencing (e.g., system failures). Forced delays are not detectable in general. As a consequence of the overhead and synchronization necessary, sequence numbers are most appropriate for smaller, closed groups.

(iii) Timestamps

Timestamps may be used to provide timeliness and uniqueness guarantees, to detect message replay. They may also be used to implement time-limited access privileges, and to detect forced delays.

Timestamps function as follows. The party originating a message obtains a timestamp from its local (host) clock, and cryptographically binds it to a message. Upon receiving a time-stamped message, the second party obtains the current time from its own (host) clock, and subtracts the timestamp received. The received message is valid provided:

1. the timestamp difference is within the *acceptance window* (a fixed-size time interval, e.g., 10 milliseconds or 20 seconds, selected to account for the maximum message transit and processing time, plus clock skew); and
2. (optionally) no message with an identical timestamp has been previously received from the same originator. This check may be made by the verifier maintaining a list of all timestamps received from each source entity within the current acceptance window. Another method is to record the latest (valid) timestamp used by each source (in this case the verifier accepts only strictly increasing time values).

The security of timestamp-based verification relies on use of a common time reference. This requires that host clocks be available and both "loosely synchronized" and secured from modification. Synchronization is necessary to counter clock drift, and must be appropriate to accommodate the acceptance window used. The degree of clock skew allowed, and the acceptance window, must be appropriately small to preclude message replay if the above optional check is omitted. The timeclock must be secure to prevent adversarial resetting of a clock backwards so as to restore the validity of old messages, or setting a clock forward to prepare a message for some future point in time (cf. Note 10.7).

10.14 Remark (*disadvantages of timestamps*) Timestamp-based protocols require that timeclocks be both synchronized and secured. The preclusion of adversarial modification of local timeclocks is difficult to guarantee in many distributed environments; in this case, the security provided must be carefully re-evaluated. Maintaining lists of used timestamps within the current window has the drawback of a potentially large storage requirement, and corresponding verification overhead. While technical solutions exist for synchronizing distributed clocks, if synchronization is accomplished via network protocols, such protocols themselves must be secure, which typically requires authentication; this leads to a circular security argument if such authentication is itself timestamp-based.

10.15 Remark (*comparison of time-variant parameters*) Timestamps in protocols offer the advantage of fewer messages (typically by one), and no requirement to maintain pairwise long-term state information (cf. sequence numbers) or per-connection short-term state information (cf. random numbers). Minimizing state information is particularly important for servers in client-server applications. The main drawback of timestamps is the requirement of maintaining secure, synchronized distributed timeclocks. Timestamps in protocols may typically be replaced by a random number challenge plus a return message.

10.3.2 Challenge-response by symmetric-key techniques

Challenge-response mechanisms based on symmetric-key techniques require the claimant and the verifier to share a symmetric key. For closed systems with a small number of users, each pair of users may share a key a priori; in larger systems employing symmetric-key techniques, identification protocols often involve the use of a trusted on-line server with which each party shares a key. The on-line server effectively acts like the hub of a spoked wheel, providing a common session key to two parties each time one requests authentication with the other.

The apparent simplicity of the techniques presented below and in §10.3.3 is misleading. The design of such techniques is intricate and the security is brittle; those presented have been carefully selected.

(i) Challenge-response based on symmetric-key encryption

Both the Kerberos protocol (Protocol 12.24) and the Needham-Schroeder shared-key protocol (Protocol 12.26) provide entity authentication based on symmetric encryption and involve use of an on-line trusted third party. These are discussed in Chapter 12, as they additionally provide key establishment.

Below, three simple techniques based on ISO/IEC 9798-2 are described. They assume the prior existence of a shared secret key (and no further requirement for an on-line server). In this case, two parties may carry out unilateral entity authentication in one pass using timestamps or sequence numbers, or two passes using random numbers; mutual authentication requires, respectively, two and three passes. The claimant corroborates its identity by demonstrating knowledge of the shared key by encrypting a challenge (and possibly additional data) using the key. These techniques are similar to those given in §12.3.1.

10.16 Remark (*data integrity*) When encipherment is used in entity authentication protocols, data integrity must typically also be guaranteed to ensure security. For example, for messages spanning more than one block, the rearrangement of ciphertext blocks cannot be detected in the ECB mode of block encryption, and even CBC encryption may provide only a partial solution. Such data integrity should be provided through use of an accepted data integrity mechanism (see §9.6; cf. Remark 12.19).

9798-2 mechanisms: Regarding notation: r_A and t_A, respectively, denote a random number and a timestamp, generated by A. (In these mechanisms, the timestamp t_A may be replaced by a sequence number n_A, providing slightly different guarantees.) E_K denotes a symmetric encryption algorithm, with a key K shared by A and B; alternatively, distinct keys K_{AB} and K_{BA} may be used for unidirectional communication. It is assumed that both parties are aware of the claimed identity of the other, either by context or by additional (unsecured) cleartext data fields. Optional message fields are denoted by an asterisk (*), while a comma (,) within the scope of E_K denotes concatenation.

1. *unilateral authentication, timestamp-based*:

$$A \rightarrow B : E_K(t_A, B^*) \quad (1)$$

Upon reception and decryption, B verifies that the timestamp is acceptable, and optionally verifies the received identifier as its own. The identifier B here prevents an adversary from re-using the message immediately on A, in the case that a single bi-directional key K is used.

2. *unilateral authentication, using random numbers*:
To avoid reliance on timestamps, the timestamp may be replaced by a random number, at the cost of an additional message:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : E_K(r_B, B^*) & (2) \end{array}$$

B decrypts the received message and checks that the random number matches that sent in (1). Optionally, B checks that the identifier in (2) is its own; this prevents a reflection attack in the case of a bi-directional key K. To prevent chosen-text attacks on the encryption scheme E_K, A may (as below) embed an additional random number in (2) or, alternately, the form of the challenges can be restricted; the critical requirement is that they be non-repeating.

3. *mutual authentication, using random numbers*:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : E_K(r_A, r_B, B^*) & (2) \\ A \leftarrow B : E_K(r_B, r_A) & (3) \end{array}$$

Upon reception of (2), B carries out the checks as above and, in addition, recovers the decrypted r_A for inclusion in (3). Upon decrypting (3), A checks that both random numbers match those used earlier. The second random number r_A in (2) serves both as a challenge and to prevent chosen-text attacks.

10.17 Remark (*doubling unilateral authentication*) While mutual authentication may be obtained by running any of the above unilateral authentication mechanisms twice (once in each direction), such an ad-hoc combination suffers the drawback that the two unilateral authentications, not being linked, cannot logically be associated with a single protocol run.

(ii) Challenge-response based on (keyed) one-way functions

The encryption algorithm in the above mechanisms may be replaced by a one-way or non-reversible function of the shared key and challenge, e.g., having properties similar to a MAC (Definition 9.7). This may be preferable in situations where encryption algorithms are otherwise unavailable or undesirable (e.g., due to export restrictions or computational costs). The modifications required to the 9798-2 mechanisms above (yielding the analogous mechanisms of ISO/IEC 9798-4) are the following:

1. the encryption function E_K is replaced by a MAC algorithm h_K;
2. rather than decrypting and verifying that fields match, the recipient now independently computes the MAC value from known quantities, and accepts if the computed MAC matches the received MAC value; and
3. to enable independent MAC computation by the recipient, the additional cleartext field t_A must be sent in message (1) of the one-pass mechanism. r_A must be sent as an additional cleartext field in message (2) of the three-pass mechanism.

The revised three-pass challenge-response mechanism based on a MAC h_K, with actions as noted above, provides mutual identification. Essentially the same protocol, called *SKID3*, has messages as follows:

$$\begin{array}{lll} A \leftarrow B : & r_B & (1) \\ A \rightarrow B : & r_A,\ h_K(r_A, r_B, B) & (2) \\ A \leftarrow B : & h_K(r_B, r_A, A) & (3) \end{array}$$

Note that the additional field A is included in message (3). The protocol *SKID2*, obtained by omitting the third message, provides unilateral entity authentication.

(iii) Implementation using hand-held passcode generators

Answering a challenge in challenge-response protocols requires some type of computing device and secure storage for long-term keying material (e.g., a file on a trusted local disk, perhaps secured under a local password-derived key). For additional security, a device such as a chipcard (and corresponding card reader) may be used for both the key storage and response computation. In some cases, a less expensive option is a passcode generator.

Passcode generators are hand-held devices, resembling thin calculators in both size and display, and which provide time-variant passwords or *passcodes* (see Figure 10.3). The generator contains a device-specific secret key. When a user is presented with a challenge (e.g., by a system displaying it on a computer terminal), the challenge is keyed into the generator. The generator displays a passcode, computed as a function of the secret key and the

ch
tic
at
re

us
ch
lo
sc
ar

tc

10.3.3

challenge; this may be either an asymmetric function, or a symmetric function (e.g., encryption or MAC as discussed above). The user returns the response (e.g., keys the passcode in at his terminal), which the system verifies by comparison to an independently computed response, using the same information stored on the system side.

For further protection against misplaced generators, the response may also depend on a user-entered PIN. Simpler passcode generators omit the user keypad, and use as an implicit challenge a time value (with a typical granularity of one minute) defined by a timeclock loosely synchronized automatically between the system and the passcode generator. A more sophisticated device combines implicit synchronization with explicit challenges, presenting an explicit challenge only when synchronization is lost.

A drawback of systems using passcode generators is, as per §10.2.1(i), the requirement to provide confidentiality for user passwords stored on the system side.

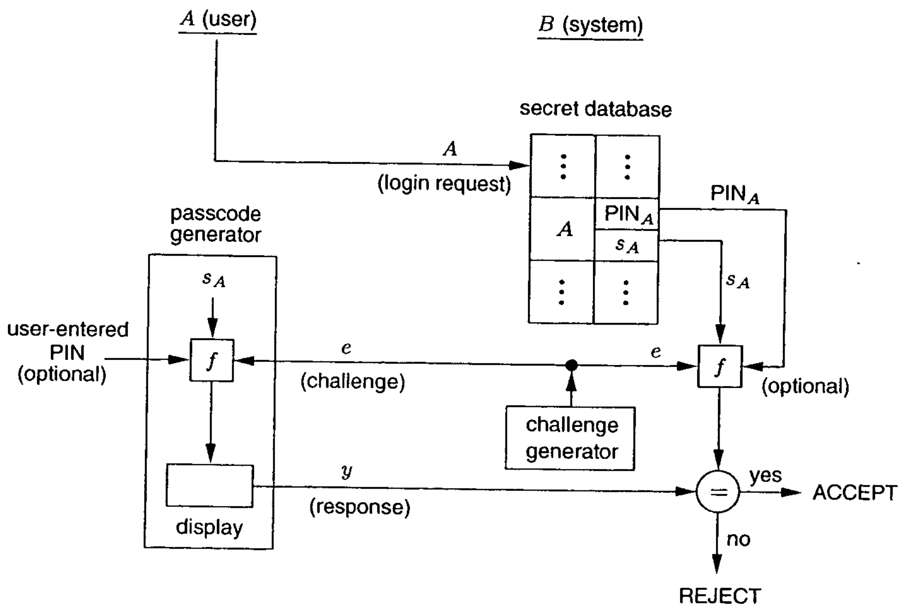

Figure 10.3: *Functional diagram of a hand-held passcode generator. s_A is A's user-specific secret. f is a one-way function. The (optional) PIN could alternatively be locally verified in the passcode generator only, making y independent of it.*

10.3.3 Challenge-response by public-key techniques

Public-key techniques may be used for challenge-response based identification, with a claimant demonstrating knowledge of its private key in one of two ways (cf. §12.5):

1. the claimant decrypts a challenge encrypted under its public key;
2. the claimant digitally signs a challenge.

Ideally, the public-key pair used in such mechanisms should not be used for other purposes, since combined usage may compromise security (Remark 10.40). A second caution is that the public-key system used should not be susceptible to chosen-ciphertext attacks,[5]

[5]Both chosen-ciphertext and chosen-plaintext attacks are of concern for challenge-response techniques based on symmetric-key encryption.

as an adversary may attempt to extract information by impersonating a verifier and choosing strategic rather than random challenges. (See Notes 8.13 and 8.58 regarding the Rabin/Williams and Blum-Goldwasser schemes.)

Incorporating a self-generated random number or *confounder* (§10.5) into the data over which the response is computed may address both of these concerns. Such data may be made available to the verifier in cleartext to allow verification.

(i) Challenge-response based on public-key decryption

Identification based on PK decryption and witness. Consider the following protocol:

$$\begin{array}{lll} A \leftarrow B: & h(r), B, P_A(r,B) & (1) \\ A \rightarrow B: & r & (2) \end{array}$$

B chooses a random r, computes the *witness* $x = h(r)$ (x demonstrates knowledge of r without disclosing it – cf. §10.4.1), and computes the challenge $e = P_A(r, B)$. Here P_A denotes the public-key encryption (e.g., RSA) algorithm of A, and h denotes a one-way hash function. B sends (1) to A. A decrypts e to recover r' and B', computes $x' = h(r')$, and quits if $x' \neq x$ (implying $r' \neq r$) or if B' is not equal to its own identifier B. Otherwise, A sends $r = r'$ to B. B succeeds with (unilateral) entity authentication of A upon verifying the received r agrees with that sent earlier. The use of the witness precludes chosen-text attacks.

Modified Needham-Schroeder PK protocol for identification. The modified Needham-Schroeder public-key protocol of Note 12.39 provides key transport of distinct keys k_1, k_2 from A to B and B to A, respectively, as well as mutual authentication. If the key establishment feature is not required, k_1 and k_2 may be omitted. With P_B denoting the public-key encryption algorithm for B (e.g., RSA), the messages in the modified protocol for identification are then as follows:

$$\begin{array}{lll} A \rightarrow B: & P_B(r_1, A) & (1) \\ A \leftarrow B: & P_A(r_1, r_2) & (2) \\ A \rightarrow B: & r_2 & (3) \end{array}$$

Verification actions are analogous to those of Note 12.39.

(ii) Challenge-response based on digital signatures

X.509 mechanisms based on digital signatures. The ITU-T (formerly CCITT) X.509 two- and three-way strong authentication protocols specify identification techniques based on digital signatures and, respectively, timestamps and random number challenges. These are described in §12.5.2, and optionally provide key establishment in addition to entity authentication.

9798-3 mechanisms. Three challenge-response identification mechanisms based on signatures are given below, analogous to those in §10.3.2(i) based on symmetric-key encryption, but, in this case, corresponding to techniques in ISO/IEC 9798-3. Regarding notation (cf. 9798-2 above): r_A and t_A, respectively, denote a random number and timestamp generated by A. S_A denotes A's signature mechanism; if this mechanism provides message recovery, some of the cleartext fields listed below are redundant and may be omitted. $cert_A$ denotes the public-key certificate containing A's signature public key. (In these mechanisms, if the verifier has the authentic public key of the claimant a priori, certificates may be omitted; otherwise, it is assumed that the verifier has appropriate information to verify the validity of the public key contained in a received certificate – see Chapter 13.) Remark 10.17 also applies here.

r and choos-
ding the Ra-

the data over
data may be

rotocol:

owledge of r
B). Here P_A
s a one-way
s $x' = h(r')$,
. Otherwise,
upon verify-
s chosen-text

edham-Schr-
s k_1, k_2 from
stablishment
-key encryp-
identification

) X.509 two-
ues based on
es. These are
ntity authen-

sed on signa-
y encryption,
; notation (cf.
mp generated
age recovery,
ert_A denotes
anisms, if the
y be omitted;
y the validity
rk 10.17 also

1. *unilateral authentication with timestamps*:

$$A \rightarrow B : cert_A, t_A, B, S_A(t_A, B) \quad (1)$$

   Upon reception, B verifies that the timestamp is acceptable, the received identifier B is its own, and (using A's public key extracted from $cert_A$ after verifying the latter) checks that the signature over these two fields is correct.

2. *unilateral authentication with random numbers*: Reliance on timestamps may be replaced by a random number, at the cost of an additional message:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : cert_A, r_A, B, S_A(r_A, r_B, B) & (2) \end{array}$$

   B verifies that the cleartext identifier is its own, and using a valid signature public key for A (e.g., from $cert_A$), verifies that A's signature is valid over the cleartext random number r_A, the same number r_B as sent in (1), and this identifier. The signed r_A explicitly prevents chosen-text attacks.

3. *mutual authentication with random numbers*:

$$\begin{array}{ll} A \leftarrow B : r_B & (1) \\ A \rightarrow B : cert_A, r_A, B, S_A(r_A, r_B, B) & (2) \\ A \leftarrow B : cert_B, A, S_B(r_B, r_A, A) & (3) \end{array}$$

   Processing of (1) and (2) is as above; (3) is processed analogously to (2).

10.4 Customized and zero-knowledge identification protocols

This section considers protocols specifically designed to achieve identification, which use asymmetric techniques but do not rely on digital signatures or public-key encryption, and which avoid use of block ciphers, sequence numbers, and timestamps. They are similar in some regards to the challenge-response protocols of §10.3, but are based on the ideas of interactive proof systems and zero-knowledge proofs (see §10.4.1), employing random numbers not only as challenges, but also as *commitments* to prevent cheating.

10.4.1 Overview of zero-knowledge concepts

A disadvantage of simple password protocols is that when a claimant A (called a *prover* in the context of zero-knowledge protocols) gives the verifier B her password, B can thereafter impersonate A. Challenge-response protocols improve on this: A responds to B's challenge to demonstrate knowledge of A's secret in a time-variant manner, providing information not directly reusable by B. This might nonetheless reveal some partial information about the claimant's secret; an adversarial verifier might also be able to strategically select challenges to obtain responses providing such information (see chosen-text attacks, §10.5).

Zero-knowledge (ZK) protocols are designed to address these concerns, by allowing a prover to demonstrate knowledge of a secret while revealing no information whatsoever (beyond what the verifier was able to deduce prior to the protocol run) of use to the verifier

U. S. DEPARTMENT OF COMMERCE
COMMISSIONER FOR PATENTS
P.O. BOX 1450
ALEXANDRIA, VA 22313-1450
IF UNDELIVERABLE RETURN IN TEN DAYS

OFFICIAL BUSINESS

02 1A
0004205065
MAILED FROM

AN EQUAL OPPORTUNITY EMPLOYER

INSUFFICIENT
ADDRESS

RECEIVED
OCT 3 1 2005
USPTO MAIL CENTER